Journal of Theoretical and Applied Information Technology
15th May 2017. Vol.95. No 9

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

COMPARATIVE ANALYSIS OF CORDIC ALGORITHM AND TAYLOR SERIES EXPANSION

[1]M. MANASA, [2]FAZAL NOORBASHA, [3]CH.L.SUDHESHNA, [4]M.SANTHOSH, [5]V.NARESH AND [6]MD. ZIA UR RAHMAN

[1, 2, 3, 4, 5, 6] Department of ECE, K L University, Vaddeswaram, Guntur, AP – India
[1]manasamuliki@kluniversity.in, [2]fazalnoorbasha@kluniversity.in, [3]sudheshnachanna@gmail.com,
[4]santhoshram786@gmail.com, [5]vaddempudinaresh@gmail.com,
[6]mdzr55@gmail.com

ABSTRACT

In recent times, power consumption and the area efficiency has become a challenge in designing Very Large Scale Integrated Circuits (VLSI). CORDIC (Coordinate Rotational Digital Computer) has come out as its solution in performing trigonometric elementary functions because of the fact that uses only shifters and adders. In this article, CORDIC potentiality is exhibited as it finds its applications in many streams like Communication, Digital Signal Processing and Image processing, angle rotation in hyperbolic, linear and circular coordinate systems, Vector transformation, scientific calculators, Robotics. The latency of Taylor series expansion to compute sine/cosine angles is also discussed. The Xilinx implementation of CORDIC is simulated and the overview of its applications in modern era is presented. The proposed paper explains the performance of CORDIC with the elimination of sine block in the initial stages as the sine of the smaller angles is negligible.

Keywords: *CORDIC, Image Processing, Sine/Cosine, Elementary Functions.*

1. INTRODUCTION

The existing research in the design of high speed and low power VLSI architectures for real-time digital signal processing (DSP) algorithms has been directed by the advances in the VLSI technology, which have provided the designers with key momentum for porting algorithm into architecture. Many of the algorithms used in DSP and matrix arithmetic require elementary functions such as division, multiplication, exponential, logarithm, trigonometric, and inverse trigonometric functions. The frequently used software solutions for the digital implementation of these functions are polynomial expansions, and lookup table method requiring number of additions/subtractions and multiplication. However, digit by-digit methods exist for the evaluation of these elementary functions, which compute faster than software solutions [1-2].

In Digital environment, the computation of trigonometric functions is quite complex and the evaluation of vector rotation consumes more time. The trigonometric sine/cosine angles can be computed using Taylor series method which involves division and multiplication operations that requires many iterations and much hardware .So, the CORDIC has come up with the beauty that it is much efficient than any other methodologies [3-4].The actual paper and the proposed paper differs in the implementation as it eliminates the sine block in the initial stages, that doesn't have any impact on the further stages because the sine for the small angles is almost zero [5-6].

The acronym of CORDIC is Coordinate Rotational Digital Computer .The main purpose of the design is to assess algebraic functions like cosine, sine, tan and inverse cosine, sine, tan angles, logarithmic functions, vector rotations and transformations. The design meets the requirements like vigorous, simple, definitive, rapid because the implementation involves only adders/subtractors and shifters[7-8].However, the angle calculation can be done more efficiently using lookup tables but the problem arises when the size of lookup tables rises. However, when the size of lookup table increases then the computation time rises and the delay shoots up. So, we prefer CORDIC [9-11]. However, the various parameters like power, delay, flip-flops, LUTs are compared by the implementation of

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

general CORDIC with the other forms of it like parallel, pipeline and generic CORDIC.

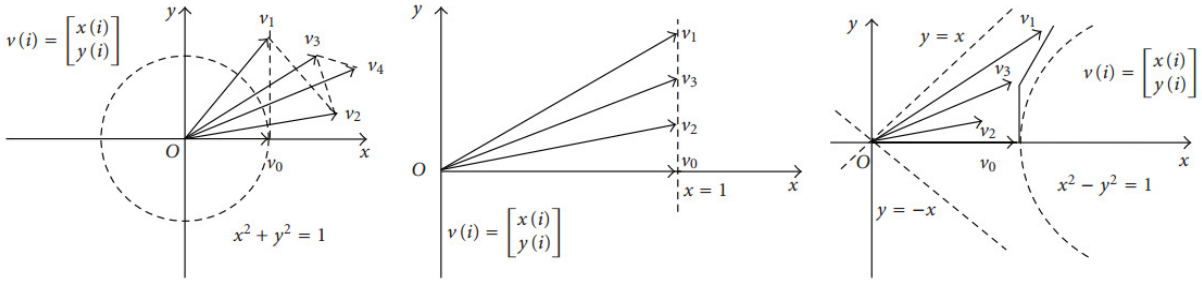
The another alternative to perform sine and cosine angle calculation is by Taylor series method but it includes many iteration steps[12] .Step1 includes assessing multiplications which require many multipliers in FPGA implementation .Step2 includes performing factorial of infinite series of numbers. The other step involves integrating these all steps by means of adders. So, we can thereby conclude that the Taylor series expansion needs much hardware when compared to other implementations. So, this problem can be eliminated by using CORDIC algorithm which uses only shifters and adders/subtractors. CORDIC

As the hardware implementation of CORDIC is considered, it uses less hardware to compute the angles, rotation and transformation of vectors thereby the iterations decreases, speed and accuracy rises. It supports the flexibility in the hardware because when the initial stages don't have impact on the computation we can even eliminate them [11].The paper elaborates the CORDIC algorithm which is performed in a simple way eliminating the ineffective sine block in the initial stages thereby the complexity, cost is reduced that implies the effective performance.

2. CORDIC ALGORITHM AND TAYLOR SERIES EXPANSION

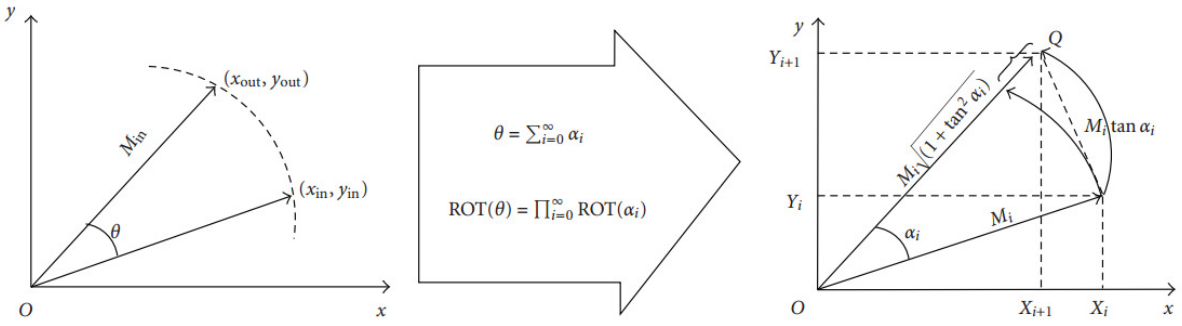
2.1 Cordic Algorithm

Even though CORDIC may not be the best technique, it is best due to its low cost implementation and efficient of a large class of applications. Quite a few designs have been projected in the literature for the CORDIC algorithm during the last two decades to provide low cost and high performance hardware solutions for real time computation of transcendental functions and two dimensional vector rotations.



a. Circular System *b. Linear System* *c. Hyperbolic System*

Figure 1: Various coordinate rotation in a system



a. b. c.

Journal of Theoretical and Applied Information Technology
15th May 2017. Vol.95. No 9

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Figure 2: 2D vector rotation in CORDIC algorithm

The CORDIC algorithms involve rotation of a vector 'v' on the XY-plane in circular, linear and hyperbolic coordinate systems depending on the function to be evaluated. Trajectories for the vector 'v_i' for successive CORDIC iterations are shown in Figure 2. This is an iterative convergence algorithm that performs a rotation iteratively using a series of specific incremental rotation angles selected so that each iteration is performed by shift and add operation. The CORDIC algorithm calculates trigonometric functions, rotation of a vector and angle of a vector by realizing two dimensional vector rotation in circular coordinate systems. Figure 2 shows the rotation of a vector with length Min by a sequence of microrotations through the elementary angles αi [12].

2.2 Taylor Series Expansion

The computation of *sinx* and *cosx* can be computed using Taylor's series expansion [10]. The series includes the multiplication and division operations which are much complex to evaluate .It involves many iteration steps which takes much computation time. The factorial for infinite numbers is to be computed. However, the hardware complexity increases because of the multipliers, adders, subtractors and dividers. However, the expansion is given as

$$\text{Sin}x = x - x^3/3! + x^5/5! - x^7/7! .....$$

Section1 describes about the introduction to CORDIC, Section 3 explains about computing sine and cosine angles using the proposed algorithm, Section 4 explains about the hardware implementation, Section 5 explains about the results and simulation, Section 6 briefs about the conclusion and then references.

3. EVALUATION OF SINE/COSINE USING CORDIC

Let us consider a vector V with coordinates (x,y) is rotated through an angle φ and the new vector obtained is V′ with coordinates (x′,y′) and the coordinates in rotational form can be given as

$$x = r\cos\theta \ , \ y = r\sin\theta \qquad (1)$$

$$V'\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x\cos\varphi - y\sin\varphi \\ y\cos\varphi + x\sin\varphi \end{bmatrix} \qquad (2)$$

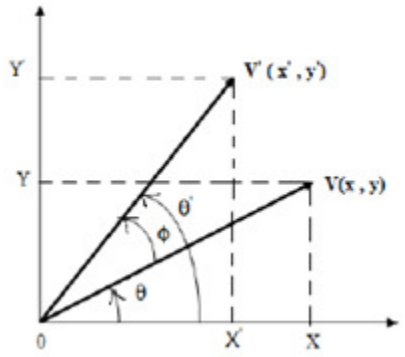


Figure 3: vector V with θ, angle of rotation φ, vector V′ with θ′

However, the above equations are derived based on the magnitude and phase of a vector V and this is illustrated by the following steps:

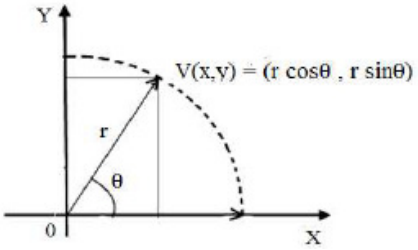


Figure 4: Magnitude r and phase θ for a vector V

Figure 3 and Figure 4 depicts about the magnitude and phase of a vector and the values of x and y are

$$x = r\cos\theta$$
$$y = r\sin\theta \qquad (3)$$

As we know that V has r as magnitude and θ as its phase angle and V′ has r as magnitude and θ′ as its phase. Now, φ can be written as below:

$$\theta' - \theta = \varphi \qquad (4)$$
$$\theta' = \varphi + \theta \qquad (5)$$

$$OX' = x' = r\cos\theta'$$
$$= r\cos(\varphi+\theta)$$
$$= r(\cos\theta.\cos\varphi - \sin\theta.\sin\varphi)$$
$$= (r\cos\theta)\cos\varphi - (r\sin\theta).\sin\varphi \qquad (6)$$

From Figure 4 and equation 3,

$OX' = x' = x\cos\varphi - y\sin\varphi$ (7)

In the same way, OY′ can be expressed as

$OY' = y' = r\sin\theta'$
$= r\sin(\varphi+\theta)$
$= r(\sin\theta.\cos\varphi+\cos\theta.\sin\varphi)$
$=(r.\sin\theta)\cos\varphi + (r.\cos\theta)\sin\varphi$
$=y\cos\varphi + x\sin\varphi$ (8)

The new vector V′ obtained from the rotation of V through an angle φ in clockwise direction can be given as

$x' = x\cos\varphi - y\sin\varphi$
$y' = y\cos\varphi + x\sin\varphi$

Similarly, value for Vector V in the clockwise direction rotating the vector V by the angle φ is

$x' = x\cos\varphi + y\sin\varphi$ (9)
$y' = y\cos\varphi - x\sin\varphi$ (10)

The equations (7), (8), (9), (10) can be expressed in the matrix from as

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\varphi & \pm\sin\varphi \\ \pm\sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

If the vector V is rotated through an angle φ in anti clockwise direction then the coordinates are

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (11)$$

In the equation (11), the matrix $\begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix}$ is called as rotation matrix R(φ) and from this matrix we can easily enumerate cosφ and sinφ. The vector rotation either in clockwise or anti clockwise can be assessed by rotating the vector $\begin{bmatrix} x \\ y \end{bmatrix}$.

The cosφ term is brought out of the matrix and the vector is still simplified as

$$R(\varphi) = \cos(\varphi)\begin{bmatrix} 1 & -\tan(\varphi) \\ \tan(\varphi) & 1 \end{bmatrix}$$

Whenever the cos term is brought out of the equation we are left with tangent angle and this in turn helps us to assess the values of sinθ and cosθ.

$x' = \cos\varphi(x - y\tan\varphi)$ (12)
$y' = \cos\varphi(y + x\tan\varphi)$ (13)

The tanφ value can be calculated by assuming the value of $\tan\varphi = 2^{-i}$ and which implies the value of φ as $\varphi = \tan^{-1}(2^{-i})$.

$$R(\varphi) = \cos(\varphi)\begin{bmatrix} 1 & -2^{-i} \\ 2^{-i} & 1 \end{bmatrix} \quad (14)$$

$$R(\varphi) = \cos(\varphi)\begin{bmatrix} 1 & -di\,2^{-i} \\ di\,2^{-i} & 1 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} \quad (15)$$

As we know that the iterations propagate and we need to evaluate for 'n' iterations.

$x_{i+1} = k_i(x_i - y_i.d_i.2^{-i})$ (16)

$y_{i+1} = k_i(y_i + x_i.d_i.2^{-i})$ (17)

where i specifies number of iterations required to reach the required angle and for instance cosφ is assumed to be a constant k_i. The value of k propagates for the further iterations so we need to integrate the values of k obtained in every stage as depicted as

$$K = \prod_{i=0}^{n-1} k_i$$

Where $\prod_{i=0}^{n-1} k_i = \cos\varphi_0 \cos\varphi_1 \cos\varphi_1 \cos\varphi_2 \cos\varphi_3 \cos\varphi_4 \ldots\ldots \cos\varphi_{n-1}$. K is the CORDIC gain and the value of k_i as

$$K = \prod_{i=0}^{7} \cos\varphi_i = \cos\varphi_0 \cos\varphi_1 \cos\varphi_2 \cos\varphi_3 \cos\varphi_4 \cos\varphi_5 \cos\varphi_6 \cos\varphi_7$$
$$= \cos45^{\circ}.\cos26.565^{\circ}.\cos14.036^{\circ}.\ldots\ldots\cos0.4469^{\circ} = 0.6073$$

Table 1 show the Different values for the calculation of gain K for 8,16,32 bit CORDIC

Table 1: Different values for the calculation of gain K for 8, 16, 32 bit CORDIC

I	$\tan(\phi_i) = 2^{-i}$	$\Phi_i = \tan^{-1}(2^{-i})$	Φ_i in radians
0	1	45	0.7854
1	0.5	26.565	0.4636
2	0.25	14.036	0.2450
3	0.125	7.125	0.1244
4	0.0625	3.576	0.0624
5	0.03125	1.7876	0.0312
6	0.015625	0.8938	0.0156
7	0.0078125	0.4469	0.0078
8	0.00390625	0.2238	0.0039
9	0.001953125	0.1119	0.0019
10	0.0009765625	0.05595	0.00098

4. DESIGN OF CORDIC HARDWARE

The design of CORDIC hardware involves the 'N' stages. Each stage corresponds to an adder/subtractor and shifters.The sgn(z_0) ,sgn($z_{1)}$.....sgn(z_n) indicates the sign bit and based on the sign of the bit it is propagated to the adjacent block and acts as adder/subtractor correspondingly. Figure 5 shows the Hardware Implementation of CORDIC.

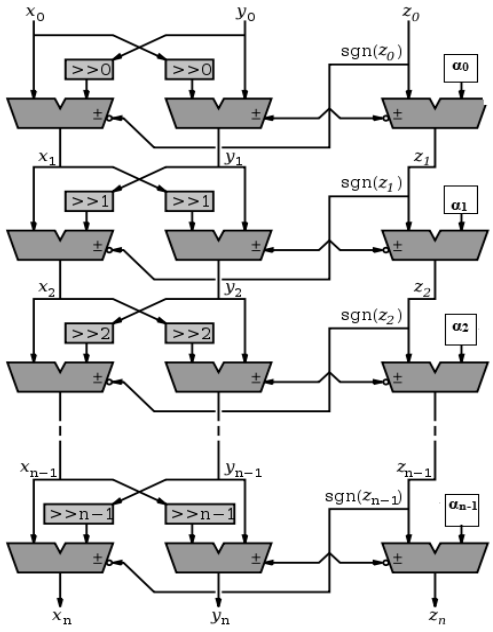


Figure 5: Hardware Implementation of CORDIC

The desired angle which has to undergo the vector rotation is loaded into Z_0,Z_1,Z_2.....Z_n bits. These bits are propagated to the next stages further. The specified angles that are to be given at every stage of vector rotation are indicated by the bits α_0, α_1, α_2 ,α_3,..... α_{n-1}.These bits corresponds to the angle values of 45°,26.565°,14.036°,....0.4469°.

The block that is to be functioned as adder/subtractor is decided based on the sign bits. Inorder to perform the vector rotation to desired angle then based on the adder/subtractor the angle is added to or subtracted from the fixed angle inorder to get the desired angle.

4.1 Implementation

This analysis could be illustrated by an example. Suppose, let us assume that vector V(x,y) should rotate to an angle of 60° , then initially the vector is at the origin and the coordinates are supposed as V(1,0) then it is rotated to an fixed angle of 45°, then the vector has to be rotated anti-clockwise and now the vector is located at 45°,and the angle that is left to reach 60° is 15°.Now,further rotation is required and the vector still has to undergo the rotation of 15° but the fixed angle in the next stage is 26.565°.So, now the vector has undergone the extra rotation and it is located at 71.565°,but the desired angle is 60°.So, now the vector has to be rotated clockwise direction inorder to tend to 60°, but the fixed angle in the next rotation is 14.036°,so it is rotated to 57.529° but we need an extra rotation of 2.471° inorder to meet 60°, but in the next step the fixed angle is 7.1° ,so it is to be rotated in the anti clockwise direction again and this process continues until the précised angle to 60° is obtained.

4.2 Advantages

1. CORDIC algorithm is very efficient, simple and robust to use because the hardware implementation uses only adders/subtractors, shifters.
2. CORDIC algorithm generates an efficient output because it uses much iteration to compute the trigonometric functions.
3. It has many real time applications in the trending technology like DSP processors, DFT and FFT algorithms, scientific calculators etc because it's simple architecture.
4. Many pipelined architectures, Multiplexer based CORDIC finds its importance in image processing because its consistency and accuracy.
5. In the hardware implementation of CORDIC such as on FPGA, the multiplexers are replaced by the logic gates, adders/subtractors, bit shifters that are simple to implement.
6. The pipelined CORDIC, parallel CORDIC, multiplexer based CORDIC, differential CORDIC are designed on the factor that to minimise the hardware and the initial stages that have no effect are eliminated.

4.3 Applications

1. The CORDIC algorithm finds its applications in signal processing, image processing, robotics, 3-D graphics, and scientific calculator.

2. It is used in communication stream for modulation techniques like AM and FM modulation, DFT and FFT computation, direct digital synthesis, adaptive filters, digital modulation techniques.
3. It is used in matrix computation, Vector interpolation and rotation, matrix decomposition, Eigen values estimation etc.
4. It is used in image enhancement, logarithmic computation and transformation, vector and image rotation.
5. It is used to operate and control the robot that has many links and joints internally, the connectivity between these links and joints and the transformation of these links is done by using vector kinematics using CORDIC.
6. It is used for vector rotation in circular, hyperbolic and linear systems, vector transformation and rotation, compute square root algorithms, evaluation of trigonometric elementary functions and to perform multiplication and division operations.

5. SIMULATION AND RESULTS

The vector v(x,y) is defined as inputs x and y as in the result and the vector coefficients after rotation are defined as xf and yf and zf specifies the desired angle to which the vector has to be rotated. The input angle is given as 10922(in degrees) and the corresponding sine angle is calculated. However, sine of the angle is calculated for the largest input in order to determine an accurate result.

The proposed architecture is coded in Verilog and synthesized using Xilinx to be implemented in Xilinx Spartan 3E device. The proposed architecture is compared with the existing CORDIC designs; the power dissipation of the proposed architecture for different clock frequencies is estimated by Xilinx XPower tool.

From the Figure 6, we can notice the RTL structure of CORDIC which defines the vector having the coordinates xi and yi, the input angle zi. The sine and cosine of the angle are defined as xf, yf and as the angle is computed in various iterative steps the zf is propagated till the end. We also can observe that the yf is zero since as we eliminated the initial stages of sine computational block, the sine value for the smaller angles is zero.

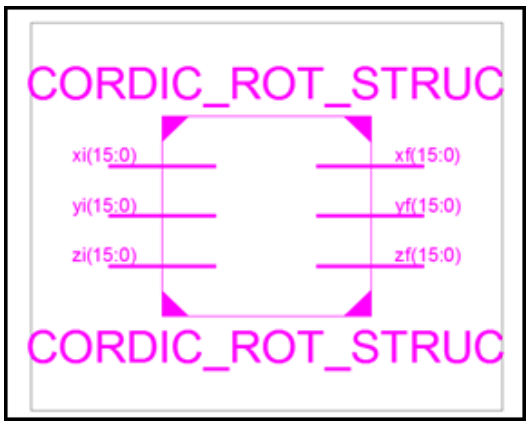


Figure 6: RTL Structure Of Sine/Cosine Wave.

We have written Verilog HDL for CORDIC architecture, Table 2 is showing that the synthesis report. We have observed the following timing analysis total REAL time to Xst completion is 8.00 secs. Total CPU time to Xst completion is 8.58 secs. Figure 7 is showing the simulation waveforms of CORDIC system.

Table2: Comparative Synthesis report analysis for CORDIC

Synthesis Report		
	Spartan 2E	Spartan 3E
Number of slice LUTs	592	503
Number of IOs	112	96
Number of Input Buffers	63	48
Number of output Buffers	63	48
Total memory usage	384112 kilobytes	212192 kilobytes
Maximum Combinational path delay	21.047ns	19.139ns

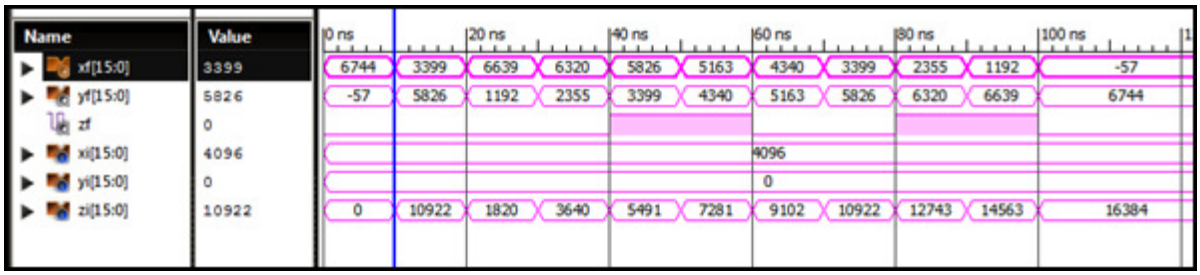


Figure 7: Verilog Simulation Results For Sine/Cosine Wave Generators.

This CORDIC system was implemented in Verilog HDL code and used two different FPGA architectures SPARTAN 2E and SPARTAN 3E. Table 2 shows the comparative synthesis report of SPARTAN 2E and SPARTAN 3E. It is showing that SPARTAN 3E is giving good performance. To get good performance need to use different design algorithms and floorplan techniques.

6. CONCLUSION

In this paper, the comparative analysis of CORDIC has been successfully implemented for 8-bit, 16-bit and 32-bit relating to speed, complexity and efficiency.CORDIC algorithm has been successfully implemented to evaluate trigonometric angles using verilog HDL programming. The CORDIC algorithm however reduces the complexity in computation of vector rotation of larger angles and the power consumption is reduced due to the flexibility in the hardware. The area efficiency is measured in terms of flip-flops, IOB's, LUTs. However, CORDIC finds its applications in the signal processing, communications, image processing, and robotics in the modern era. CORDIC has been compared with different Taylor series expansion and also the Look up Tables and observed that the latency is degraded to 50% due to decrease in number of iterations. In terms of Resolution it is observed that the 32-bit CORDIC is best suited. However, the proposed methodology is best suited in comparison with other algorithms because of the reason that it eliminates the initial stages thereby reducing the complexity and cost which implies increasing the speed. The proposed CORDIC processor has 17% lower slice-delay product with a penalty of about 13% increased slice consumption on Xilinx Spartan 3E device.

REFERENCES

[1] V.Naresh, B.Venkataramani and R.Raja, "An Area efficient multiplexer based CORDIC", International Conference on Computer Communication and Informatics (ICCCI-2013), Jan.04-06, 2013, Coimbatore, INDIA.

[2] T. Nareshkumar, K.Venkataramana Reddy, "Implementation of CORDIC algorithm for FPGA based computers using verilog HDL",International Journal of Industrial Electronics and Electrical Engineering, Volume-2,Issue10,Oct-2014, PP. 1-5.

[3] KavyaSharat, Dr.B.V.Uma, Sagar D.M, "Calculation of Sine and Cosine of an Angle using the CORDIC algorithm", International Journal of Innovative Technology and Research (IJITR)Volume No.2,Issue No.2,Feb-March 2014,PP.891-895.

[4] RanjitaNaik, Riyazahammad Nadaf, "Sine-Cosine Computation using CORDIC Algorithm", International Journal of Advanced Research in Computer and Communication Engineering,Vol.4,Isssue 9,September 2015, PP. 44-48.

[5] Rohit Shukla , Kailash Chandra Ray, "Low Latency Hybrid CORDIC algorithm, Computers IEEE Transactions on Computers,Vol. 63, Issue 12, Dec. 2014, pp. 3066-3078.

[6] R.K.Jain, B.Tech Thesis, "Design and FPGA Implementation of CORDIC-based 8-point 1D DCT Processor", NIT Rourkela, Orissa, 2011.

[7] K.Y.Chang, Y.N.Zeng, P.Chen, Z.P.Qin, "Application of CORDIC algorithm in sine & cosine function and its implementation on FPGA", Computer Engineering and Applications, Vol.49, No.7, 2013, PP.140-143.

[8] P.V.Kumar, C.S.Pari, G.R.Murthy, E.K.Wong, "Low Energy Improved Speed and High Throughput CORDIC cell to Improve Performance of Robots Processor", Asian Journal of scientific Research, Vol.8, No.3, 2015, PP.381-391.

[9] X.W.chen, G.X.Liu, W.M.Tang, "Arbitary logarithm CORDIC algorithm based on expansion convergence domain and FPGA implementation", China Measurement & Test, Vol.41, No.7, 2015, PP.108-111.

[10] L.X.Deng, J.S.An, "A low Latency High-Throughput Elementary Function Generator Based on Enhanced Double Rotation CORDIC", IEEE symposium on Computer Applications and Communications (SCAC), 2014, PP.125-130.

[11] Junwei Li , Jiandong Fang , Bajin Li , Yudong Zhao, "Study of CORDIC algorithm based on FPGA ", Control and Decision Conference (CCDC), 2016 Chinese, 28-30 May 2016, DOI: 10.1109/CCDC.2016.7531747

[12] Mario Garrido, Petter Källström, Martin Kumm, and Oscar Gustafsson, "CORDIC II: A New Improved CORDIC Algorithm", IEEE Transactions on Circuits and Systems—II: Express Briefs, Vol. 63, No. 2, February 2016. PP. 186-190.